ALP LAB

AI Inference on Ethos-U and DRP-AI3

AN-008 · E1M-AEN (Ethos-U) and E1M-X V2N (DRP-AI3)

Document Number: AN-008 **Revision:** 0.2 **Date:** June 2026 **Status:** Preliminary

alplab.ai

1 Scope

Take a trained classification or detection model (TensorFlow / PyTorch), compile it for the target NPU, deploy it, and measure inference latency end-to-end on:

- **Ethos-U55** on the E1M-AEN family (E3– E8 silicon).
- **DRP-AI3** on the E1M-X V2N / V2N-M1 families.

Both NPUs are vision-friendly (CNN-class operators); LLM-class workloads on V2N-M1 use the on-module DeepX DX-M1 instead and are covered in **AN-009**.

Audience	ML engineers shipping vision-AI firmware to fielded SoMs.
Prerequisites	A trained model (Keras / TFLite / ONNX), familiarity with `west build`, host with Python 3.10+ for the model-compile toolchain.
Outcome	Compiled model runs on the target NPU; firmware prints per-frame inference time; image-classification accuracy matches the floating-point reference within the post-training-quantisation budget.
Time	60– 90 minutes (model-compile is the long pole the first time).
Source	`docs/tutorials/16-inference-mobilenet.md` and `examples/camera-vision/ai-object-detection-realtime/` in alp-sdk.

Table 1 Scope summary

2 Model Compile

2.1 Ethos-U55 (AEN)

The Ethos-U55 needs models pre-compiled to its custom **Ethos-U command stream** via the Arm `vela` compiler:

```
pip install ethos-u-vela
vela --accelerator-config ethos-u55-256 \
     --system-config Ethos_U55_Embedded \
     --output-dir ./compiled \
     ./model.tflite
```

The output `model_vela.tflite` is the file you embed in the firmware. The SDK's unified `<alp/inference.h>` dispatcher loads it through its Ethos-U backend.

2.2 DRP-AI3 (V2N)

The DRP-AI3 uses Renesas' **DRP-AI Translator** tool:

```
docker run --rm -v $PWD:/work renesas/drp-ai-translator:v1.5 \
    --input /work/model.onnx \
    --output /work/compiled/ \
    --target v2n
```

The output bundle contains the executable image + a weights file; both are loaded by the same `<alp/inference.h>` dispatcher through its DRP-AI backend.

> **Note:** Both compilers expect models **already** quantised to INT8 (post-training quantisation with a representative dataset). Float-32 models are not supported on either NPU; the compiler will reject them.

3 Software Walkthrough

```c
#include "alp/inference.h"

extern const uint8_t model_blob[];       // Vela- or DRP-AI-compiled, xxd -i'd
extern const size_t  model_blob_len;

int main(void)
{
    alp_inference_t *inf = alp_inference_open(&(alp_inference_config_t){
        .backend     = ALP_INFERENCE_BACKEND_AUTO,    // Ethos-U on AEN, DRP-AI on V2N
        .model_data  = model_blob,
        .model_size  = model_blob_len,
        .format      = ALP_INFERENCE_MODEL_VELA,      // _DRPAI on V2N
        .arena_bytes = 512 * 1024,                    // MobileNetV2 224 quant
    });
    if (inf == NULL) {
        printk("[inf] open failed: last_err=%d\n", (int)alp_last_error());
        return 1;
    }

    /* The SDK owns the input/output tensor buffers; fill the input
     * in place each frame, then invoke. */
    alp_inference_tensor_t in;
    alp_inference_get_input(inf, 0, &in);

    while (1) {
        camera_capture_rgb888(in.data);   // see AN-003 for the camera bring-up

        uint32_t t0 = k_cycle_get_32();
        alp_inference_invoke(inf);        // blocks on the NPU until the result lands
        uint32_t t1 = k_cycle_get_32();

        alp_inference_tensor_t out;
        alp_inference_get_output(inf, 0, &out);
        const uint8_t *scores = (const uint8_t *)out.data;

        int     idx  = argmax_u8(scores, out.shape[out.rank - 1]);
        uint8_t conf = scores[idx];
        uint32_t dt_us = k_cyc_to_us_floor32(t1 - t0);

        printk("[inf] class=%d score=%u dt=%uus\n", idx, conf, dt_us);
        k_msleep(100);
    }

    alp_inference_close(inf);
}
```

Build + flash with:

```
west build -b <BOARD> examples/camera-vision/ai-object-detection-realtime
west flash
```

The same `<alp/inference.h>` API works on both NPUs; with `ALP_INFERENCE_BACKEND_AUTO` the SDK dispatches to the appropriate backend based on the SoM SKU declared in `board.yaml`.

4 Expected Output

```
[inf] open: model 612 KiB, backend ETHOS_U (variant U55)
[inf] tensor arena 512 KiB allocated
[inf] class=287 score=222 dt=8423us
[inf] class=287 score=227 dt=8418us
[inf] class=287 score=220 dt=8431us
```

Class 287 is "lynx" in the ImageNet label set. The example bundles a small `labels.txt` so the firmware can also print the human-readable name. The score is the raw quantised (uint8) class confidence read straight from the output tensor.

Typical inference times (MobileNetV2 224 × 224 INT8):

Target	Time (typ.)	Notes
Ethos-U55, 256 MACs (E3 / E7)	TBD ms	Single inference, no double-buffering.
Ethos-U55 + U85 (E4 / E6 / E8)	TBD ms	Two-NPU parallel execution.
DRP-AI3 (V2N / V2N-M1)	TBD ms	Single inference.

Table 2 Inference latency benchmark

5 Troubleshooting

Symptom	Likely cause / fix
`vela` rejects the model	Op not supported. Check the Ethos-U55 supported-op list at review.mlplatform.org. Fall back to CPU for unsupported ops.
DRP-AI Translator errors on conv stride	DRP-AI3 only supports strides 1 and 2. Modify the model to use compatible stride values.
Inference returns all zeros	Quantisation calibration dataset was unrepresentative; re-quantise with a sample of real input.
Latency 10× expected	Model fell back to CPU; one or more ops not NPU-accelerated. Check the SDK's compile log.
Out-of-memory on Ethos-U55	`alp_inference_invoke` returns `ALP_ERR_NOMEM`. Increase `arena_bytes` in the `alp_inference_config_t` (or `inference.default_arena_kib:` in `board.yaml`); Vela's `network_summary_*.csv` report gives the exact size the compiled model needs.

Table 3 Common failures

6 References

- **Canonical tutorial**: `docs/tutorials/16-inference-mobilenet.md` in alp-sdk.
- **Examples**: `examples/camera-vision/ai-object-detection-realtime/`, `examples/aen/edgeai-vision-aen/`, `examples/ai/ai-anomaly-detection-vibration/`.
- **SDK API**: `<alp/inference.h>` (unified dispatcher), `<alp/model.h>` (`.alpmodel` package parser), `<alp/backend.h>` (backend registry).
- **Backend registry**: `docs/architecture/backend-registry.md`.
- **Companion**: **AN-003** (MIPI CSI camera input), **AN-009** (DeepX M1 for > 4 TOPS workloads).

7 Revision History

Revision	Changes	Date
0.1	Initial draft.	May 2026
0.2	Aligned to the current alp-sdk: replaced the non-existent `<alp/ai.h>` / `<alp/ai/ethos_u.h>` / `<alp/ai/drp_ai.h>` API with the unified `<alp/inference.h>` dispatcher (`alp_inference_open` / `get_input` / `invoke` / `get_output` / `close`) and added `<alp/model.h>` / `<alp/backend.h>` references. Fixed example paths into their topic folders (`examples/camera-vision/`, `examples/ai/`) and added `examples/aen/edgeai-vision-aen/`. Updated arena-OOM guidance to `arena_bytes` / `default_arena_kib` and refreshed the expected-output listing.	June 2026

Table 4 Revision History